Exam 2 Review

CS 2308
Spring 2015

Jill Seaman

Exam 2

- Friday, March 27
- In class, closed book, closed notes, clean desk
- 15% of your final grade
- 50 minutes to complete it
- I recommend using a pencil (and eraser)
- All writing will be done on the test paper I will hand out.
- No calculators.

Exam Format

- 100 points total, 4 pages
  - Writing programs/functions/code
  - Multiple choice
  - Fill-in-the-blank/short answer
  - Tracing code (show what is the output)

Which Lectures?

- Chapter 9
- Chapter 10
- Chapter 13

Ch 9: Pointers

- Address operator (&)
- Pointer variables: how to define (data type)
- Dereferencing operator (*)
- Pointers and arrays
  - an array variable is the address of its first element
  - array[index] = *(array + index)
- Pointer arithmetic (if ptr points to a var of type d):
  - ptr + n = address in ptr + n * sizeof(d)
- Initializing Pointers

Ch 9: Pointers, cont.

- Comparing pointers
- Pointers as function parameters
  - Pass by reference using pointers as parameters
  - Pointers used as parameters accepting arrays as arguments
- Dynamic memory allocation
  - new operator
  - new with arrays
  - delete
  - return pointers from functions (duplicateArray)

Ch.10: Strings and Things

- Character testing + conversion
  - isalpha, isdigit, isupper, islower, isspace
  - toupper tolower
- C-strings
  - '\0' -terminated, data type = char array
- C-strings: library functions: (know what they do)
  - strlen
  - strcpy (assignment)
  - strcmp (test, comparison)

Ch.10: Strings and Things (cont.)

- Predefined string class
  - how to define and initialize string variables
- operations:
  - =, <<, >>, relational ops, [n]
- member functions
  - at(n)
  - length() and size()
  - append(str)
- know how to use these to write code!

Ch.13: Classes

- Procedural programming
  - What it is
  - Main problem: (handling changes to impl)
- Object oriented programming:
  - What it is
  - How it solves problem of procedural programming
  - **encapsulation**
  - **data (or information) hiding**
  - **interface**
  - class vs instance (object)

Ch.13: Classes (cont)

- Declaring a class:
  - Members: variables and functions
  - private vs public, access rules
  - syntax: class declaration
  - syntax: member function definitions
  - accessor/mutator (getter/setter), const function
  - How to define instances (variables, objects).
  - How to access members (dot operator)
- Separating specifications from implementation
  - What goes where (class decl, member func defs)
  - How to compile

Ch.13: Classes (cont)

- Inline member functions
- Constructors
  - How to name, return type?
  - When are they called? what do they do?
  - Default constructor
  - Passing arguments to constructors
- Destructors
  - What it is, how to name it, when is it called?
- Overloaded constructors and member functions

Example Tracing Problem

What will the EXACT output of the following program be?

```
int main () {
   int *ptr1, *ptr2;
   int foo1;

   foo1 = 42;
   ptr1 = &foo1;
   *ptr1 = 13;
   ptr2 = ptr1;

   cout << "foo1 - " << foo1 << endl;
   cout << "*ptr1 - " << *ptr1 << endl;
   cout << "*ptr2 - " << *ptr2 << endl;

   int x[] = {1,2,3};
   ptr1 = &x[1];
   *ptr2 = *(x+1);

   cout << endl;
   cout << "*ptr1 - " << *ptr1 << endl;
   cout << "*ptr2 - " << *ptr2 << endl;
}
```

Example Programming Problem

Consider a Rectangle class, which stores a rectangle using two floating point values: width and height.

It has 2 constructors (one default, one with 2 arguments), and accessors and mutators for the width and height.

The default rectangle has width and length equal to 1. The other constructor takes the initial width and height. The rectangle also has a function area() that calculates the area of the rectangle.

(a) Write the class declaration.

(b) Implement all the class functions.

How to Study

- Review the slides
  - understand all the concepts, quiz yourself
- Use the book to help understand the slides
  - there will be no questions over material (or code) that is in the book but not on the slides
- Review programming assignments (fix yours!)
  - get printouts of solutions in my office
- Try some exercises from the book
- Practice, practice, practice
- Get some sleep